In the Claims

This listing of claims will replace all prior versions and listings of claims in the application:

1. (Currently Amended) A processor having a changeable architected state, comprising:

instruction memory for storing instructions;

an instruction pipeline, wherein an instruction which passes entirely through the pipeline alters the architected state and wherein the pipeline comprises circuitry for fetching instructions from the instruction memory into the pipeline;

~~circuitry~~ an annul word memory for storing an annul code having a plurality of annul bits, ~~an~~ each annul bit ~~corresponding to each~~ having a one-to-one correspondence to one instruction of a group of instructions in the pipeline; and

circuitry for preventing one or more selected instructions in the group from altering the architected state in response to the corresponding annul bit of the annul code.

Claims 2 to 4. (Canceled)

5. (Currently Amended) The processor of claim ~~3~~ 1:

wherein the instruction pipeline further comprises a plurality of execution units;

wherein the plurality of execution units are operable such that in a given clock cycle an integer number N of the plurality of execution units are scheduled to execute; ~~and~~

wherein the circuitry for preventing one or more selected instructions in the group from altering the architected state comprises circuitry for coupling annul bits to respective ones of the plurality of execution units; and

wherein the circuitry for coupling the annul bits to respective ones of the plurality of execution units comprises circuitry for coupling only the integer number N of the annul bits to the plurality of execution units which are scheduled to execute in the given clock cycle whereby on an immediately following clock cycle annul bits beginning at an N+1 annul bit are coupled to execution units scheduled to execute of that following cycle.

Claim 6. (Canceled)

7. (Currently Amended) The processor of claim 5:

wherein the group of instructions corresponding to the annul code comprise instructions corresponding to a software loop scheduled to execute for an integer M number of iterations; and

wherein during a given iteration the circuitry for preventing prevents one or more of the group of instructions corresponding to the annul bits of the annul code from altering the architected state in response to the annul bits of the annul code and the annul code based on a relationship of the given iteration to the integer M number of iterations preventing differing instructions from altering the architected state during different iterations.

Claim 8 and 9. (Canceled)

10. (Currently Amended) The processor of claim 1 wherein the annul code is generated in response to one or more constant generating instructions and loaded into the annul word memory.

11. (Currently Amended) The processor of claim 1 wherein the annul code is loaded into the annul word memory from a memory.

12. (Currently Amended) The processor of claim 1 wherein the annul code is an immediate value in an immediate operand instruction passing through the pipeline loaded into the annul word memory in response to execution of the immediate operand instruction.

Claims 13 to 15. (Canceled)

16. (Currently Amended) The processor of claim 1:

wherein the annul code is loaded into the annul word memory from a selected location of memory in response to an instruction having a condition predicate;

wherein the annul code comprises a first annul code stored in a first location in the memory loaded into the annul word memory in response to the condition predicate being satisfied; and

wherein the annul code comprises a second annul code stored in a second location in the memory loaded into the annul word memory in response to the condition predicate not being satisfied.

17. (Currently Amended) The processor of claim ~~16~~ 1:

and further comprising a first data register and a second data register;

wherein the annul code is loaded into the annul word memory from a selected one of the first data register and the second data register in response to an instruction having a condition predicate;

wherein the annul code comprises a first annul code ~~is~~ stored in the first data register loaded into the annul word memory in response to the condition predicate being satisfied; and

wherein the annul code comprises a second annul code ~~is~~ stored in the second data register loaded into the annul word memory in response to the condition predicate not being satisfied.

18. (Currently Amended) The processor of claim ~~16~~ 1:

and further comprising a data register;

wherein the annul code is loaded into the annul word memory from a selected half of the data register in response to an instruction having a condition predicate;

wherein the annul code comprises a first annul code ~~is~~ stored in a first one-half of the data register loaded into the annul word memory in response to the condition predicate being satisfied; and

wherein the annul code comprises a second annul code ~~is~~ stored in a second one-half of the data register different from the first one-half loaded into the annul word memory in response to the condition predicate not being satisfied.

Claim 19. (Canceled)

20. (Currently Amended) The processor of claim 1:

and further comprising a register;

wherein the register stores the annul code which comprises a set of annul bits having a first logical value and a set of annul bits having a second logical value;

wherein the annul code is loaded into the annul word memory from the register in response to an instruction having a condition predicate;

wherein the circuitry for preventing prevents instructions corresponding to annul bits having a first logical value from altering the architected state in response to the condition predicate being satisfied; and

wherein the circuitry for preventing prevents instructions corresponding to annul bits having a second logical value opposite the first logical state from altering the architected state in response to the condition predicate not being satisfied.

21. (Currently Amended) The processor of claim 1 and further comprising circuitry for storing a portion of the annul code into the annul word memory in response to receipt of an interrupt.

22. (Currently Amended) A method of data processing comprising the steps of:

identifying at compile time prior to execution a group of instructions including a tree of a plurality of conditional branch instructions;

at compile time prior to execution for each conditional branch instruction within the group of instructions

forming a first annul code having an annul bit corresponding to each instruction following the conditional branch instruction, the annul bit having a first logical state for instructions following the detected conditional branch instruction executed if a condition of the conditional branch instruction is satisfied and a second logical state opposite to the first logical state for instructions following the detected conditional branch instruction executed if the condition of the conditional branch instruction is not satisfied,

forming a second annul code having an annul bit corresponding to each instruction following the conditional branch instruction, the annul bit having the second logical state for instructions following the detected conditional branch instruction executed if a condition of the conditional branch instruction is satisfied and the first logical state for instructions following the detected conditional branch instruction executed if the condition of the conditional branch instruction is not satisfied;

upon execution of the group of instructions

detecting each conditional branch instruction,

evaluating the condition of the conditional branch instruction,

loading the corresponding first annul code if the condition of the conditional branch instruction is satisfied,

loading the corresponding second annul code if the condition of the conditional branch instruction is not satisfied,

executing each instruction following the conditional branch instruction if the corresponding annul bit of the corresponding annul code has the first state, and

not executing each instruction following the conditional branch instruction if the corresponding annul bit of the corresponding annul code has the second state.